DSGRID Documentation

****

********** Overview of the CDAT interface to dsgrid **********

****

INTRODUCTION TO NGMATH

The ngmath library is a collection of interpolators and approximators for one−dimensional, two−dimensional
and three−dimensional data. The packages, which were obtained from NCAR, are:

natgrid −− a two−dimensional random data interpolation package based on Dave Watson's nngridr.

dsgrid −− a three−dimensional random data interpolator based on a simple inverse distance weighting algorithm.

fitgrid −− an interpolation package for one−dimensional and two−dimensional gridded data based on Alan Cline's Fitpack. Fitpack uses splines under tension to interpolate in one and two dimensions.

csagrid −− an approximation package for one−dimensional, two−dimensional and three−dimensional random
data based on David Fulker's Splpack. csagrid uses cubic splines to calculate its approximation function.

cssgrid −− an interpolation package for random data on the surface of a sphere based on the work of Robert Renka. cssgrid uses cubic splines to calculate its interpolation function.

shgrid −− an interpolation package for random data in 3−space based on the work of Robert Renka. shgrid uses a modified Shepard's algorithm to calculate its interpolation function.

COMPARISION OF NGMATH PACKAGES

Three−dimensional packages −− shgrid, csagrid and dsgrid.

shgrid is probably the package of choice for interpolation. It uses a least squares fit of biquadratics to construct its interpolation function. The interpolation function will pass through the original data points.

csagrid uses a least squares fit of cubic splines to calculate its approximation function: the calculated surface will not necesarily pass through the original data points. The algorithm can become unstable in data
sparse regions.

dsgrid uses a weighted average algorithm and is stable in all cases, but the resultant interpolation is not usually smooth and execution time is very slow. dsgrid is probably best used when csagrid and shgrid

fail or for comparative purposes.

Two−dimensional packages −− natgrid, fitgrid, csagrid and dsgrid.

natgrid is the package of choice in most cases. It implements a very stable algorithm and has parameters
for adjusting the smoothness of the output surface.

fitgrid offers user−settable parameters for specifiying derivatives along the boundary of the output grid which are not available in natgrid.

csagrid produces an approximate two−dimensional surface which may be smoother than that produced by fitgrid
and natgrid.

dsgrid is not recommended for two−dimensional surfaces. natgrid is superior in all respects.

One−dimensional packages −− fitgrid and csagrid.

fitgrid is definitely the package of choice. It has many features not available in csagrid, such as interpolating parametric curves, finding integrals, handling periodic functions, allowing smoothing that
varies from linear to a full cubic spline interpolation and specifying slopes at the end points.

Interpolation on a sphere −− cssgrid.

cssgrid is designed specifically for interpolating on a sphere. It uses cubic splines to calculate an interpolation function.

DSGRID PACKAGE

dsgrid implements a simple weighted average interpolation algorithm. The input for the interpolation is a set
of randomly spaced two or three−dimensional coordinates with functional values at those coordinates; the output
is a set of interpolated values at the user specified coordinates. The coordinates may be in a list or a gridded
format. The coordinates in the output grid must be monotonically increasing in each coordinate direction, but need
not be evenly spaced.

dsgrid uses a simple weighted average method to do its interpolation, the value of the weights being inversely
proportional to the distance form an input coordinate to the point where the interpolated value is desired. As
the default, the influence varies as the the inverse cube of the distance. The sum of the weights is normalized
to unity and an interpolated value is the sum of the the products of the known functional values and the
calculated weights. The capability exits to vary the power of the distances used to compute the weights. For
powers greater than one, the interpolated surface is flat at the input data: the size of the spot increases with

increasing values of the power. If the power of the distances is one, the weights are computed as the linear
inverse distances and the areas around input data are cone shaped. If the power of the distances is less than one,
the areas around input data form a cusp at the data points.

DSGRID CONTENTS

Access through Python to the dsgrid package from NCAR's ngmath distribution is provided directly through the module
dsgridmodule.so which was generated as a Python C language extension in order to export the functions from the
original C language library

REQUIRED FILE

dsgridmodule.so –– the Python interface to the ngmath dsgrid package.

USEFUL FILES

ds.py –– the object oriented interface including a general help package.
dsgridtest.py –– the code to test ds.py and to write documentation.

USAGE

This module is designed to use in two ways. One is through the use of the object oriented interface to the underlying
functions. This approach is recommended for users not already familiar with the original dsgrid distribution because
it simplifies the calls to the routines. The other method uses the original functions calling them directly from Python.

––
––––––––––––––––––– OBJECT ORIENTED APPROACH ––––––––––––––––

The ds module contains the Dsgrid class and the single method rgrd

STEP 1.

Rectangular Gridded Output Coordinates

To make an instance, r, type:

import ds

r = ds.Dsgrid(xi, xo, yi, yo)

or

r = ds.Dsgrid(xi, xo, yi, yo, zi, zo)

where xi, yi and zi are the input coordinate arrays in list format
while xo, yo and zo are the and output grid coordinate arrays which
must be increasing but not necessarily evenly spaced.

List Format Output Coordinates

To make an instance, r, type:

```
import ds

r = ds.Dsgrid(xi, xo, yi, yo, griddedOutput = 'no')

or

r = ds.Dsgrid(xi, xo, yi, yo, zi, zo, griddedOutput = 'no')
```

where xi, yi and zi are the input coordinate arrays in list format
and xo, yo and zo are the output coordinate arrays in list format.

DSGRID has regridding functions which carry out their respective interpolations in single or double precision. The choice is determined by the type of the coordinate arrays submitted in makking the instance.

STEP 2.

Gridded output or list output is accessed with the same call to the method function, rgrd.
The choice is made in the creation of an instance of the Dsgrid class in STEP 1.

Type
dataOut = r.rgrd(dataIn) where

dataIn –– input data in list form
dataOut –– output data in gridded or list form

––
––––––––––––––––––– ORIGINAL FUNCTION APPROACH –––––––––––––––––

The module dsgridmodule.so exports the following functions to Python from the original C library:

Single precision procedures:

grid2s – primary function for gridding 2D data.
grid3s – primary function for gridding 3D data.
seti – set int parameter values.
geti – retrieve values for int parameters.
setr – set float parameter values.
getr – retrieve values for float parameters
setc – set char parameter values.
getc – retrieve values for char parameters.
pnt2s – interpolate 2D data at specified individual points.
pnt3s – interpolate 3D data at specified individual points.

Double precision procedures:

grid2d – primary function for gridding 2D data.
grid3d – primary function for gridding 3D data.
setrd – set float parameter values.
getrd – retrieve values for float parameters
pnt2d – interpolate 2D data at specified individual points.
pnt3d – interpolate 3D data at specified individual points.

Information on the use of the routines is available by importing dsgridmodule and printing the docstring
of interest. For example, documentation for the routine grid2s is obtained by typing

```
import dsgridmodule

print dsgridmodule.grid2s.__doc__
```

This same information is available in the help package.

```
import ds

print ds.help('grid2s')
```

A description of the control parameters is not in the dsgridmodule documentation. It can be found by typing

```
import ds
ds.printParameterTable()
```

A hard copy of the full documentation on the use of each of the routines is written to the file dsgrid.doc after
importing ds by typing

```
ds.document()
```

DOCUMENTATION

Documentation is provided through Python's docstrings, essentially Python style program comments. A help package
provides instructions on the use of dsgrid. A table of contents is printed to the screen by typing

```
ds.help()
```

after importing ds.

A hard copy of this documentation is written to the file dsgridmodule.doc after import dsgridtest by typing

```
dsgridtest.document()
```

TESTING

To run a test of the 2D and 3D interpolations and to get a copy of this documentation, type

cdat dsgridtest.py

HELP PACKAGE EXAMPLE

********** Default Parameter Table **********

name	type	legal values	default	description
dmv	float	any	–9999.	special value for data init and use with dmx parameter
dmx	float	> 0.	1.0e20	for weights, use only input data within distance dmx
erf	char	any file name	"sterror"	error file
exp	float	> 0.	3.0	power to use for inverse distances in computing weights
shd	int	0 = no or 1 =	0	controls whether the shadowing feature is on